Insect Level Intelligence is Sufficient to Move Furniture

Bimal Sharma[1] **and Frederic Maire**[1]
Queensland University of Technology,
School of Electrical Engineering and Computer Science
bimalprakash.sharma@hdr.qut.edu.au,f.maire@qut.edu.au

Abstract

Collaborations to solve complex tasks, which are not achievable by a single individual, are frequently observed in nature. Collaborative behaviours can be exhibited by insects who have very limited intelligence and communication abilities. Collaboration can occur on the basis of pure reactive behaviours where individuals of a team react only to their current percepts. These behaviours are usually not learned. For example, in the case of ants, they are encoded in the genome.

In this paper we show that a small set of simple reactive behaviours is sufficient to enable a team of robots to complete complex collaborative tasks in a domestic environment. Our research work does not include any learning capabilities. We demonstrate an emergent complex team behaviour with two identical robots sharing the same set of basic behaviours. The two robots are able to locate and move a piece of furniture inside a room.

http://www.araa.asn.au/acra

1 Introduction

The behaviour of social insects has inspired the design of many cooperative robotic systems. Cooperation between social insects occurs although they have individually little intelligence [Hayashi *et al.*, 2012]. Social insects use their cognitive abilities to collaborate and perform complex tasks including food gathering and nest building. For example, termites build nests and complex mounds without any central leadership. The termites join sites where the population of the site is actively increasing and where there is active digging or excavation propensity [Green *et al.*, 2010]. Cognitive abilities can be created in robotics by designing suitable frameworks to implement behaviours [Finio *et al.*, 2010]. In the context of this paper, *insect level intelligence* implies absence of planning and very limited memory requirement.

Inspired by the natural behaviours of insects, we have designed a collection of simple reactive behaviours that enable humanoid robots to perform relatively complex behaviours like moving a piece of furniture (in our experiments, a small table). The behaviours are implemented as nested automata where each node corresponds to a behaviour and transitions are triggered by events.

There are three main types of behaviours in the robotic literature, namely, deliberative, reactive and hybrid [Baklouti *et al.*, 2010]. Deliberative behaviours have a planning step between sensing and acting based on a global map of the environment. Reactive behaviours rely only on current sensor data to decide what action to take next [Adouane *et al.*, 2010]. In this paper, we show that a reactive behaviour framework provides enough expressiveness for two robots to collaborate to move a piece of furniture. More complicated tasks, like moving all the tables located in a room to one of its walls, can also be accomplished within this framework. In a similar spirit, a termite-inspired system based on low-level rules was introduced in [Werfel *et al.*, 2010] to enable a swarm of robots to construct large structures.

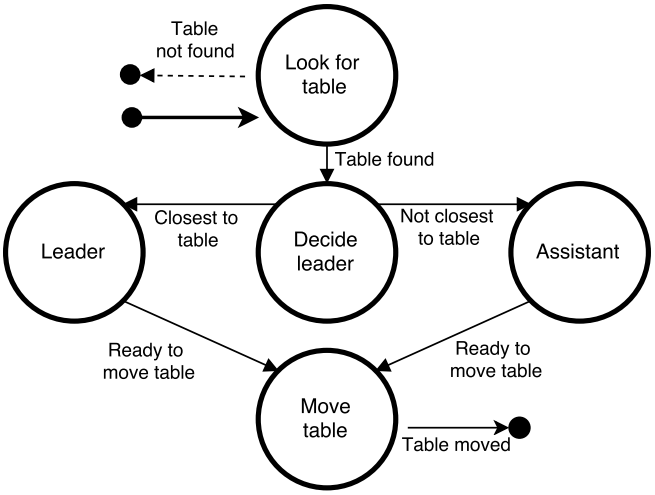

Figure 1: High level complex behaviours performed by the two robots. The vertices of the graph represent behaviours. Labels on arcs connecting behaviours indicate events.

2 Related Work

Behaviour-based robotics is a paradigm for composing primitive behaviours into more complex behaviours [Brooks, 2010], [Bradford *et al.*, 2010]. Robots exhibit complex behaviours despite having little knowledge of their immediate environment [Nicolescu *et al.*, 2010].

Humanoid robots can successfully carry out a wide range of tasks to assist humans [Dawood *et al.*, 2010]. Research studies have led to demonstrations such as cooking [Gravot *et al.*, 2010], cleaning [Yamazaki *et al.*, 2010], folding clothes [Nozawa *et al.*, 2010], handing over objects smoothly and carrying heavy objects [Murooka *et al.*, 2010]. Furthermore, the natural human harvesting behaviour observed in agriculture can be emulated by humanoid robots [Yaguchi *et al.*, 2010]. Humanoids can pick soft and delicate crops like tomatoes, strawberries and apples autonomously by using simple behaviours [Martin *et al.*, 2010]. Robots can imitate and perform actions by observing the motion patterns performed by other robots or humans [Dawood *et al.*, 2010]. The cognitive models used to perform the tasks mentioned above are defined by programming a set of rules [Rosiery *et al.*, 2010].

We use a reactive architecture design in our framework. One such framework is the flexible behavior engine (FlexBE) [Romay *et al.*, 2010]. FlexBE is a framework for composing, executing, and supervising hierarchical finite-state machines (HFSMs) for high-level tasks in robotic systems. The framework is an extension of the SMACH task execution framework [Bohren *et al.*, 2010]. FlexBE provides a graphical user interface for composing and executing behaviors. FlexBE focuses on human-robot collaboration through adjustable autonomy [Schillinger , 2010].

Tools for implementing agent and multi agent behaviors using hierarchical state machines have been used extensively in RoboCup [Risler *and Stryk*, 2008]. One such tool is Agent Behavior Specification Language (XABSL). XABSL is based on a pragmatic and formal approach. XABSL has a modular behavior architecture based on concurrent hierarchical finite state machines and a specification language used for describing hierarchical state machines. Using XABSL, multiple agents can synchronise and perform a behaviour at the same time. If an agent tries to perform a behaviour, it waits for the other agents to be ready before performing the behaviour.

The design of hierarchical behaviors has been made easier with graphical interfaces. These tools are widely used in RoboCup for building hierarchical finite state machines (HFSM) [Hagel *et al.*, 2006].

We implemented failure prevention in our framework. Robots can work together to recover from failure by detecting failure and collaborate to performing actions to prevent the failure from occurring [Schillinger *et al.*, 2010].

To explore which domestic chores can be completed with reactive behaviours, we consider the task of moving a piece of furniture. More specifically, we consider the task of moving a table. If the table is small and light, the task can be completed by a single robot. For heavier or more bulky tables, cooperation between multiple robots is required.

In the next section, we discuss how to decompose the complex behaviour into simpler behaviours.

3 Moving a Heavy Table (Two Robots)

Each robot is controlled by a predefined set of behaviours. The top behaviour depends on the task to be completed as illustrated in Figure 1. To move the table, the robots first need to find the table.

3.1 *Look for table*

Once the robot is switched on, it calls the behaviour ***Look for table*** shown in Figure 1. Similar behaviours have been introduced for robots looking for a ball in robot soccer [Risler *and Stryk*, 2008]. It is not guaranteed that the table will initially be in the field of view of the robot. Therefore, the robot has to look for it. There are no pre-conditions for this behaviour. This behaviour is time-limited. When it stops the post-conditions are as follows. Either the robot is facing the table or it failed to find the table within the allocated limit. If the table has been found, the table is in the middle of the field of view. In order to locate the table, the robot looks left, straight ahead and right by turning its head. If the table is not seen, the robot spins 180 degrees on the spot and looks left and right to locate the table. If the table is still not found, this robot walks randomly for about 1 metre and looks again. If the ***Look for table*** behaviour completes successfully, the robot switches to another behaviour to approach the table.

3.2 Collaboration between the leader and assistant

In order to simplify the coordination between agents and prevent collisions, we break the symmetry between the two robots. A leader and an assistant roles are assigned at run time. Because of this, it can be argued that the behaviour of the robot is not purely reactive. However the leader/assistant role is the only information memorized.

The leader/assistant roles are determined during a protocol handshake between the two robots. Each robot broadcasts an availability message. The *distance to the table* is sent with the first broadcast message. The distance from the robot to the blue table is a function of the position of the blue blob in the image. The robot that is closest to the table becomes the leader while the other robot takes the role of the assistant. If there is a tie, the robot with the lower IP address is selected as the leader. The assistant is the first to act. It moves away from the table and the leader. Once the assistant has made its move, it sends a *moved away* message to the leader confirming that the way has been cleared for the leader to move to the table. Upon receiving the *moved away* message from the assistant, the leader aligns itself to the table by calling the behaviours ***Re-centre to middle of the table*** and ***Align to the middle of the table*** the shown in Figure 3.

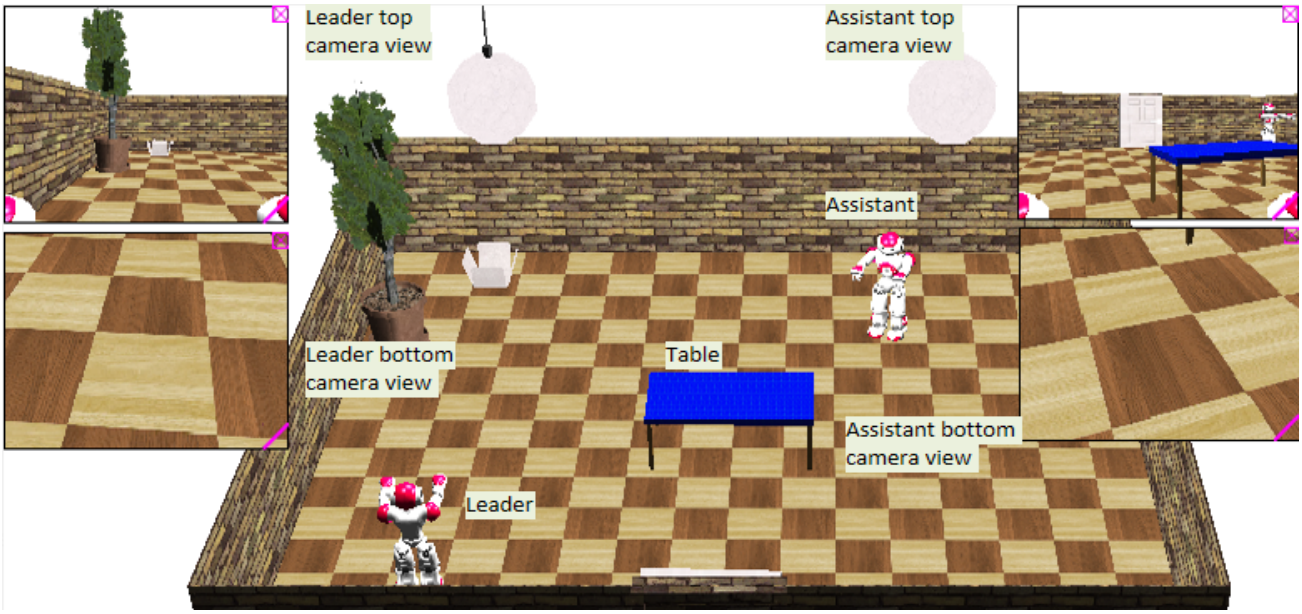

Figure 2: Experimental setup. The two robots are driven by the same set of reactive behaviours. The table in the room is moved towards a wall as a result of the emerging behaviour.

Message	Agent(s)	Direction
Distance to table	Leader, Assistant	Both ways
Out of the way	Assistant	To Leader
Ready to lift	Leader, Assistant	Both ways

Table 1: Messages exchanged between the leader and the assistant

After the leader has aligned itself to the table, a message is sent to the assistant to signal the successful positioning. When the assistant receives this message, it can start moving to the table to align itself to the opposite side of the table.

3.3 Collision avoidance

When the robots start on the same side of the table, they run the risk of colliding with each other while navigating to the table. To prevent collisions, the leader activates the behaviour ***Wait until clear way*** shown in Figure 4. The pre-condition of this behaviour is that the assistant is on the same side of the table as the leader. The post-condition is the assistant has cleared the way for the leader. If the leader and the assistant are on the same side of the table, the assistant moves out of the way, then sends a message *clear way* to the leader. The leader is then ready to go to the closest corner of the table.

3.4 Finding the longer side of the table

The robot needs to be positioned at the corner of the table so that it can later assess which side of the table is the longer by simply looking left and right. To get to the corner of the table the robot invokes the behaviour ***Go to table***. The robot targets the nearest corner of the table. The pre-condition of this behaviour is that the table is in the field of view of the robot. The post-condition is that the corner of the table is in the central bottom region of the field of view as shown in Figure 5. That is, the robot is close to a corner of the table.

After determining which side adjacent to the corner is the longest, the robot performs a walk sideways (like a crab) to position itself at the middle of the longest side of the table. This behaviour is named ***Position to centre of table side***. The pre-condition is that the robot is standing close to a corner of the table. The post-condition is that the robot is standing close to the middle of the long side of the table. During this behaviour, the robot spins at an angle to align its shoulders parallelly to the table. The robot then walks sideways to move toward the middle of the table. The robot ends up with its shoulders aligned parallelly to the longer side of the table. Upon completion of this behaviour, the robot is in a position to put its arms under the table and lift it.

3.5 Preventing the table from slipping and falling off the arms

If the robots do not lift the table synchronously, then the table might fall from their arms. To prevent this situation, the robots signal each other to synchronize and lift the table at the same time. The leader sends a *ready to move table* signal to the assistant. The assistant terminates the behaviour ***Wait until leader at centre of table*** when it receives the *ready to move table* signal. The pre-condition is that the assistant has cleared way for the leader. The post-condition is that the leader has aligned itself close to the table. Upon receiving the *ready to lift* message from the leader, the assistant activates the ***Go to opposite side of table*** behaviour. The assistant is now ready to move to the opposite side of the leader. The

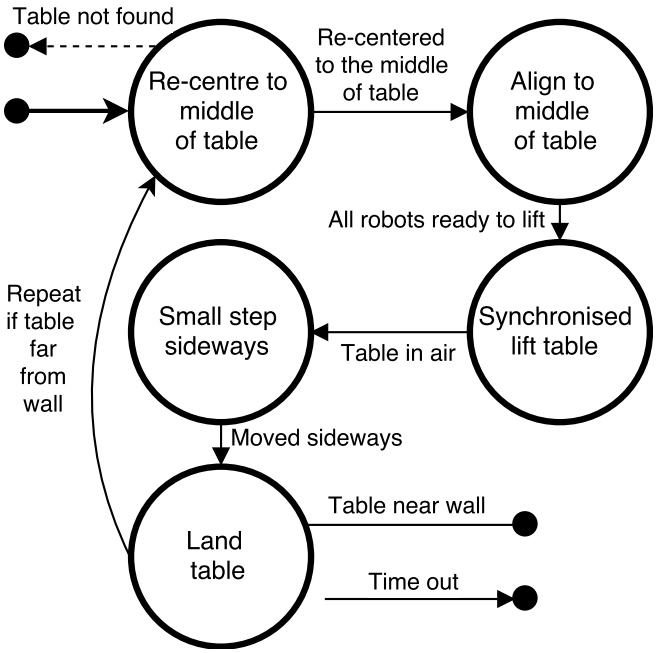

Figure 3: States of the *Move table* behaviour. This behaviour invokes five other behaviours represented by the nodes of the automaton. The solid arcs correspond to the successful termination of auxiliary behaviours whereas the dashed arcs indicate a failure exit condition. The solid circle represents the timer of the whole complex behaviour. In order to move the table safely, the robots cycle through the five behaviours.

pre-conditions are the table is in the field of view of the robot and the leader is positioned at a long side of the table. The post-condition is the assistant is positioned opposite to the leader across the table. The assistant can detect whether the leader stands behind the table by checking whether the image blob of the table is below the image blob of the leader.

The two robots are positioned opposite each other across the table and are ready to move the table when the assistant arrives at its target position opposite to the leader. The assistant sends a signal when it arrives. Once the signal is received, the leader can signal the start of the behaviour ***Synchronised lift table.***

The ***Move table*** behaviour shown in Figure 3 is essentially the same for the leader and the assistant apart from the signalling part as the leader orchestrates the motion. The pre-condition of this behaviour is that the robot is standing at the long side of the table opposite to the other robot. The post-condition is the table has been moved close to the wall. The robot looks toward the wall and compares the top edge of the table facing the wall. The robot is close to the wall when the top edge of the table is in line with the bottom edge of the wall from the field of view of the robot as shown in Figure 6.

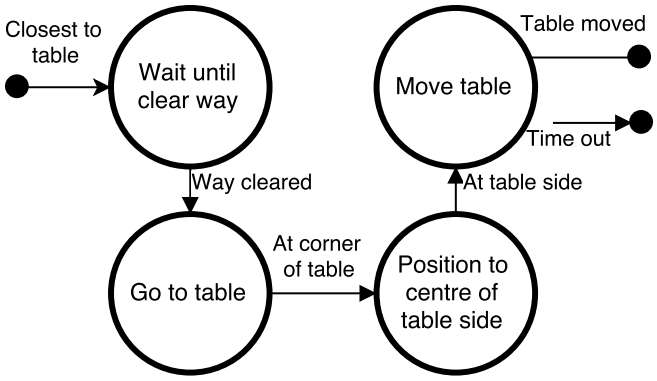

Figure 4: Decomposition of the behaviour of the *Leader*. The leader is the robot closest to the table. The leader waits for the assistant to send the *Way cleared* signal before calling the behaviour *Go to table*. Once the behaviour *Position to centre of table side* has completed successfully, the leader sends the *At table side* message to the assistant. The leader is then ready to lift the table.

3.6 Moving with the table

Moving with the table on their arms is difficult for the robots due the following two problems. The robots may not be aligned properly at the table, therefore their paths may diverge with the number of steps they take sideways. This will cause the table to fall off their arms. Reciprocally, the robots may converge as the number of steps increases causing the robots to walk closer and push each other with the table resulting in one or both robots falling.

To solve this problem, the behaviour ***Move table*** relies on the repeated iteration of five other behaviours as shown in Figure 3. These behaviours are ***Re-centre to middle of table***, ***Align to middle of table***, ***Synchronised lift table***, ***Small step sideways*** and ***Land table***. These behaviours reduce the chances of the table falling by adjusting regularly the relative positions of the robots before they lift again the table. We discuss the behaviours next.

The behaviour ***Align to middle of table*** aligns the shoulders of the robot parallel with the edge of the table within a tolerance of 20 degrees. This prevents the two robots converging to or diverging from the table during movement. The behaviour ***Small step sideways*** ensures that the robot takes just enough steps to avoid the table falling from the arms of the robot or the robot colliding its body with the table.

In the next section, we present further details about the behaviour framework we implemented.

4 Behaviour Framework for Simple Domestic Tasks

We call *behaviour* any function running on a robot that

- has optional inputs and possibly a timer.

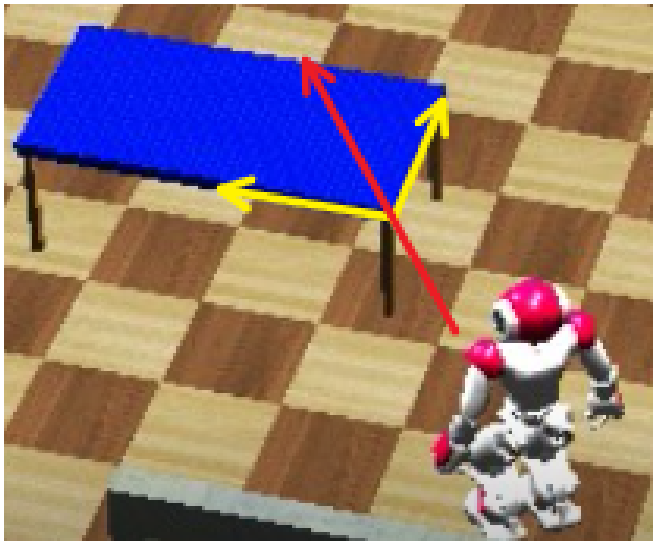

Figure 5: The robot positions itself relative to a corner of the table so that the edges of the table marked with yellow arrows form a *V* shape in its field of view. The red arrow should appear vertical in the field of view of the robot. Once this position is reached, the robot can look left and right to determine the longer side.

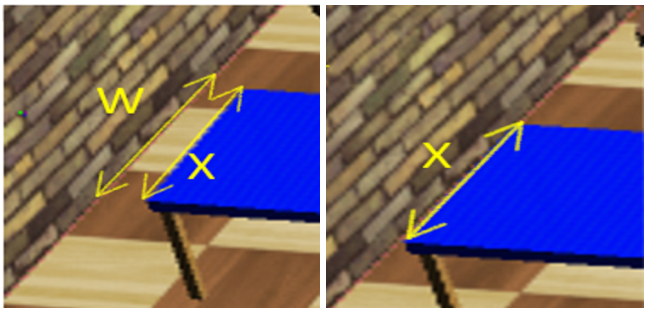

Figure 6: Field of view of the robot when aligning the table close to the wall. The image on the left shows the field of view of the robot when the left edge of the table has not aligned to the edge of the floor. X is the edge of the table and W is the edge of the floor. The image on the right shows the field of view of the robot when the edge of the table and the edge of the floor have aligned. The table has been moved to the wall.

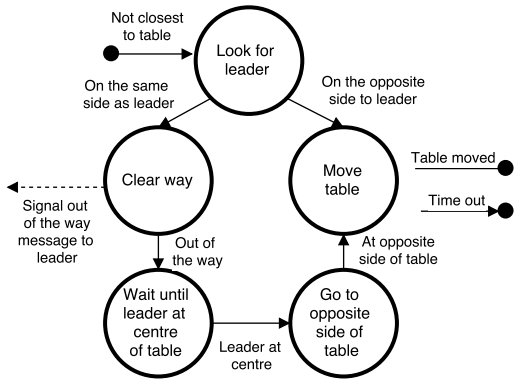

Figure 7: Decomposition of the behaviour *Assistant* from Figure 1. The assistant waits for the leader to be in position at the table before moving itself to the opposite side of the table.

- returns a *termination status* and optional output values.

In our implemented framework, behaviours are blocking calls. That is, when a behaviour (caller) starts another behaviour (callee), the caller waits until the termination of the callee. The callee returns some status (success, failure, time-out) and optionally some more information, for example, results from an image processing algorithm. We define two types of behaviours; complex behaviours and primitive behaviours. A *complex behaviour* calls on other behaviours to complete a task sequentially. An example of a complex behaviour calling other behaviours sequentially is the behaviour ***Synchronised lift table*** shown in Figure 8 and explained in Behaviour 1. The termination condition of this complex behaviour is successful when both robots have lifted the table synchronously. Failure is declared when either robot fails to lift the table or when the timer has timed out. Collaboration emerges between the robots when they perform these behaviours.

A behaviour is said to be *primitive* with respect to a library of behaviours if this behaviour does not call other behaviours. Primitive behaviours can be sophisticated. For example, a primitive behaviour can involve the execution of simultaneous threads. The behaviour ***Synchronised Lift table*** appearing in Figure 8 is a primitive behaviour that performs the task of lifting a table synchronously after messages have been exchanged successfully between the two robots.

Behaviours might fail or potentially trap the agent in a repeating pattern. To address this problem, a timer is used as a watchdog to prevent deadlocks and endless loops. The timer value is an optional input argument of the behaviour function. Once the timer has expired, the behaviour completes and the time-out status is returned back to the caller. For instance, the complex behaviour ***Move table*** in Figure 3 contains a timer. The timer can time out during the execution of any behaviour of Figure 3.

We have created a library of behaviours sufficiently rich to move a table from one point to another. From a programming point of view, the library of behaviours is a set of abstract classes. We implemented a class for each behaviour. The methods of these classes send instructions to the robot simulator for the robot to perform actions.

As an illustration of the framework, the implementation of the primitive behaviour ***Align to middle of table*** is shown in Figure 9 and outlined in Behaviour 2. In this behaviour, the robot rotates on the spot looking for the table. If the table is

found, the robot adjusts its angular pose to face the closest corner of the table. There are two possible outcomes for this behaviour. If the behaviour terminates successfully, the robot will be facing the table. Otherwise, if the behaviour fails, the robot will have simply moved to a different position.

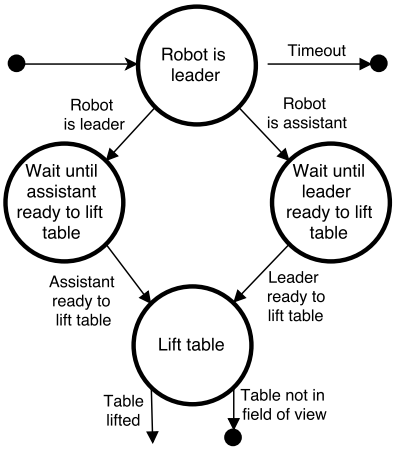

Figure 8: Decomposition of behaviour ***Synchronised lift table*** as shown in Behaviour 1. The leader and assistant poll every second to check if the messages have been sent and received for the table to be lifted.

Behaviour 1 Synchronized lift table

```
if is leader then
    wait until assistant is ready to lift table
    lift table
else
    signal leader ready to lift table
    wait for acknowledgement signal
    lift table
end if
```

Behaviour 2 Align to middle of table

```
look left and right to determine which side is the longest
while the difference between the lengths of the left and
      right sides is too large do
    align body parallelly to the table
    walk sideways towards the longer side for a small
      distance
    look left and right to determine which side is the
      longest
end while
```

The robots sense their environment through the two cameras located in their head. One camera is forward-looking, the other is downward-looking. Standard image processing techniques are applied to identify regions of interest in the images taken by the cameras of the robots. The resolution of all images was set to 640×480 pixels. For the behaviour-based control of the NAO robots, we used the state machine implementation of FlexBE [Romay *et al.*, 2010].

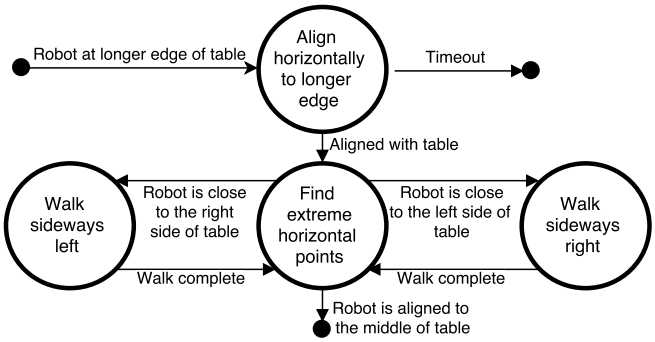

Figure 9: Decomposition of the behaviour ***Align to middle of table*** as shown in Behaviour 2. The robot looks on both sides to determine if it has aligned to the middle of the table.

5 Experiments

All experiments were conducted in the Webots robot simulator [Tllez *et al.*, 2010]. Webots is a development environment used to program and simulate mobile robots. Webots uses the NaoQi distributed object framework simulator to send commands to the Nao Robot.

We designed the scene as a room with a table placed in the middle (see Figure 2). The robots were placed at different selected locations each time a trial was run as shown in Figure 10. The goal of this experiment was for two robots to collaborate and move the table to a wall. The robots had to move the table close to a wall as in a scenario of re-arranging a room.

5.1 Simulation results

In our experiments, the time-out for each task was set to 35 minutes. If the task was not completed within that time limit, the experiment was recorded as a failure. The behaviour ***Look for table*** complete successfully all the time as the table is easy to find in the room used for the experiments. The robots were able to collaborate, synchronize and move the table close to the wall regardless of their starting position. The ***Go to table*** behaviour completed successfully in an average of 30 seconds.

The leader was able to determine the longer side of the table due to the accurate positioning achieved by the behaviour ***Position to centre of table side***. A large amount of time was taken to adjust to the corner of the table. An example of the field of view of the robot when adjusting to the closest corner of the table is shown in Figure 5.

As expected, the leader and assistant aligned to the middle of the table within the specified error margin. Reducing the error margin increases the time for the behaviour ***Align to the middle of the table*** to complete. The table did not fall or slip off the arms of the robot in all trials when the relative difference between the left and right sides was kept below 20% in the loop condition of Behaviour 2.

Aligning the shoulders to the table as parallel as possible

Behaviour start position	Look for table	Elect leader	Leader	Wait until clear way	Go to table	Position to centre of table side	Go to the lifting point	Assistant	Look for leader	Clear way	Wait until leader at centre of table	Go to opposite side of table	Move table	Re-centre to middle of table	Align to middle of table	Synchronised Lift Table	Small step sideways	Land table	Total (seconds)	Average of TOTAL (mins)
Robot standing opposite to each other long side	6	8	572	0	25	133	414	784	1	0	572	212	233	56	112	12	40	13	1603	26.7
Robot starting ahead and behind of each other	6	8	728	156	25	133	414	940	1	156	572	212	233	56	112	12	40	13	1915	31.9
Robots starting facing away from the table	112	8	600	28	25	133	414	812	1	28	572	212	233	56	112	12	40	13	1765	29.4
Robots starting facing the table	0	8	572	0	25	133	414	784	1	0	572	212	233	56	112	12	40	13	1597	26.6
Robots starting on the same side	6	8	728	156	25	133	414	940	1	156	572	212	233	56	112	12	40	13	1915	31.9
Robots starting opposite to each other short side	6	8	572	0	25	133	414	784	1	0	572	212	233	56	112	12	40	13	1603	26.7
Average of all behaviours	**23**	**8**	**629**	**57**	**25**	**133**	**414**	**841**	**1**	**57**	**572**	**212**	**233**	**56**	**112**	**12**	**40**	**13**	**1733**	**28.9**

Figure 10: This table shows the time taken by the robot to complete each behaviour in seconds based on different starting positions. A total of 60 trials were performed.

increased the number of steps that the robot could take sideways from 4 steps to 10 steps. The robots were able to take about 10 steps sideways without dropping the table or colliding their body with the table. The error margin used was 20 degrees.

The average amount of time taken to complete the whole task is 29 minutes. Most of the time is spent waiting for the other robot to complete a behaviour. In particular, the behaviours ***Wait until clear way***, ***Wait until leader at centre of table*** and ***Wait until ready to lift table*** require one robot to wait until the other robot has completed its behaviour and sent a signal. By eliminating these behaviours, the total time to complete the task can be reduced to 12 minutes. However, without these behaviours the chance of a collision happening is 30%.

Failure rate

The time required to complete the tasks increases when the error margin is reduced. There is a trade-off between the completion time of the tasks and the failure rate. For example, when the robot adjusts itself to face the closest corner of the table, an error margin of 20% is used. The center of the field of view is located horizontally at position 320 (in pixels). Therefore the target horizontal position for the closest corner is in the range $[256, 384]$. We observed that reducing the 20% error margin to 5% doubled the time it took the robot to complete the task. The relationship between the failure and the maximum time allowed is plotted in Figure 11.

Error margins for robust performance of behaviours

To reduce the probability of failure, multiple experiments were run to adjust error margins for the individual behaviours. The adjustment process for each behaviour starts with 20% error margin and is reduced by 1% with each trial until the behaviour is *robust* (that is, does not fail in 20 trials). Behaviours have error margins between 5% and 20% depending on the amount of accuracy needed to achieve robustness.

Our experimental results show that the robots avoided collision with the table by stopping when the lowest pixel of the table was seen more than 80 percent to the bottom of the field of view of the robot.

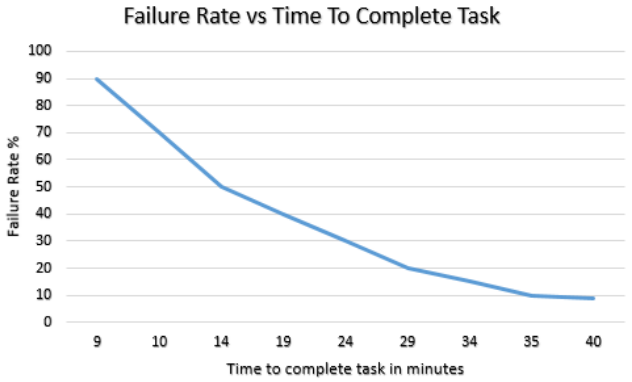

Figure 11: Plot of the failure rate versus the maximum time allowed to complete the task. The failure rate decreases with maximum time allowed.

6 Conclusion

In this paper we have experimentally demonstrated that a relatively complex task, namely moving a table, that requires cooperation between two robots can be completed within a reactive behaviour framework. We have shown that with a minimum amount of signalling, coordination of the movements of the robots can also be achieved.

Swarm of reactive robots are more robust to individual failures as all the robots are identical. Although the overall team behaviour of the robots is not optimal time-wise, this approach leads to simple and robust solutions that have potential applications in agricultural and service robotics. The set of behaviours we have introduced can be extended to allow a group of robots to clear an entire room of its chairs and tables.

Our experiments have so far been limited to manipulating a single rectangular table. In future work, we will demonstrate that the same strategy is suitable for more complex tasks like setting a room for a class. That is, putting in place tables and chairs according to a target pattern.

References

[Hayashi *et al.*, 2012] Y Hayashi, M Yuki, K Sugawara, T Kikuchi and K Tsuji Rhythmic behavior of social insects from single to multibody. *Robotics and Autonomous Systems, vol. 60, no. 5, pp. 714-721, 2012.*

[Finio *et al.*, 2010] B. Finio and R. Wood Distributed power and control actuation in the thoracic mechanics of a robotic insect. *Bioinspir. Biomim., vol. 5, no. 4, pp. 6, 2010.*

[Baklouti *et al.*, 2010] Emna Baklouti, Nader Ben Amor, Mohamed Jallouli Reactive control architecture for mobile robot autonomous navigation. *Robotics and Autonomous Systems, Volume 89, March 2017, Pages 9-14,ISSN 0921-8890, https://doi.org/10.1016/j.robot.2016.09.001.*

[Adouane *et al.*, 2010] L. Adouane, A. Benzerrouk, P. Martinet Mobile robot navigation in cluttered environment using reactive elliptic trajectories. *in: 18th IFAC World Congress 44 (1), 2011, pp. 1380113806*

[Brooks, 2010] R. A Brooks Intelligence Without Representation. *Artificial Intelligence 47 (1991)139-159. MIT Press, Cambridge, Massachusetts, 1985.*

[Bradford *et al.*, 2010] A. Towle Jr. Bradford, Monica Nicolescu An auction behavior-based robotic architecture for service robotics. *Intelligent Service Robotics (2014) 7:157174.*

[Nicolescu *et al.*, 2010] Monica and Mataric:scheme Monica N. Nicolescu and Maja J. Mataric A Hierarchical Architecture for Behavior Based Robots. *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems Bologna, ITALY, July 15-19, 2002, 227-233.*

[Martin *et al.*, 2010] Francisco Martin, Carlos E. Aguero, Jose M. Canas A Simple, Efficient, and Scalable Behavior-based Architecture for Robotic Applications. *Advances in Intelligent Systems and Computing, volume 418, pages 611-622, 2016*

[Yaguchi *et al.*, 2010] Hiroaki Yaguchi, Kotaro Nagahama, Takaomi Hasegawa, Masayuki Inaba Development of an autonomous tomato harvesting robot with rotational plucking gripper. *Intelligent Robots and Systems (IROS) 2016 IEEE/RSJ International Conference on, pp. 652-657, 2016, ISSN 2153-0866.*

[Gravot *et al.*, 2010] Gravot, F., Haneda, A., Okada, K., Inaba, M. Cooking for humanoid robot, a task that needs symbolic and geometric reasonings. *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International* Conference on, pp.462,467.

[Yamazaki *et al.*, 2010] Yamazaki, K., Ueda, R., Nozawa, S., Mori, Y., Maki, T., Hatao, N., Okada, K., Inaba, M. System integration of a daily assistive robot and its application to tidying and cleaning rooms. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, pp.1365,1371.*

[Nozawa *et al.*, 2010] Nozawa, S., Kakiuchi, Y., Okada, K., Inaba, M. Controlling the planar motion of a heavy object by pushing with a humanoid robot using dual-arm force control. *Robotics and Automation (ICRA), 2012 IEEE International Conference on, pp.1428,1435.*

[Murooka *et al.*, 2010] Murooka, M., Noda, S., Nozawa, S., Kakiuchi, Y., Okada, K., Inaba, M. Manipulation strategy decision and execution based on strategy proving operation for carrying large and heavy objects. *Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp.3425,3432.*

[Romay *et al.*, 2010] Romay, A., Kohlbrecher, S., Stumpf, A., von Stryk, O., Maniatopoulos, S., Kress-Gazit, H., Conner, D. C. (2017). Collaborative autonomy between high-level behaviors and human operators for remote manipulation tasks using different humanoid robots: Collaborative autonomy between high-level behaviors. *Journal of Field Robotics, 34(2), 333-358. doi:10.1002/rob.21671*

[Bohren *et al.*, 2010] BOHREN, J., COUSINS, S. The SMACH High-Level Executive *IEEE Robotics Automation Magazine 17 (2010), Nr. 4, p. 1820. ISSN 1070-9932, 2010.*

[Schillinger , 2010] Schillinger, P. (2015). An approach for runtime-modifiable behavior control of humanoid rescue robots. *Technische Universitat Darmstadt.*

[Schillinger *et al.*, 2010] Schillinger, P., Kohlbrecher, S., and von Stryk, O. (2016). Human-robot collaborative high-level control with an application to rescue robotics. *In IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden (pp. 27962802). doi: 10.1109/ICRA.2016.7487442.*

[Rosiery *et al.*, 2010] Rosiery S. Maia, Luiz M. G. Gonalves *Intellectual Development Model for Multi-Robot Systems.* L.M.G. J Intell Robot Syst (2015) 80: 165. doi:10.1007/s10846-015-0224-06

[Dawood *et al.*, 2010] F. Dawood and C. Loo *Incremental episodic segmentation and imitative learning of humanoid robot through self-exploration.* Neurocomputing, vol. 173, no. 3, pp. 1471-1484, 2016.

[Tllez *et al.*, 2010] R. Tllez, C. Angulo Webots Simulator 5.1.7 *Artificial Life, vol. 13, no. 3, pp. 313-318.*

[Green *et al.*, 2010] Green B, Bardunias P, Turner JS, Nagpal R, Werfel J. Excavation and aggregation as organizing factors in de novo construction by mound-building termites.

Proceedings of the Royal Society of London (2017) B 284: 20162730. http://dx.doi.org/10.1098/rspb.2016.2730

[Werfel *et al.*, 2010] Werfel, J., Petersen, K. and Nagpal, R. Designing Collective Behavior in a Termite-Inspired Robot Construction Team. *SCIENCE (2014), vol. 343, no. 6172, pp. 754-758.*

[Risler *and Stryk*, 2008] M. Risler and O. von Stryk. Formal Behavior Specification of Multi-Robot Systems Using Hierarchical State Machines in XABSL. *AAMAS08-Workshop on Formal Models and Methods for Multi-Robot Systems*

[Hagel *et al.*, 2006] V. Hugel et al. Specifications and Design of Graphical Interface for Hierarchical Finite State Machines. *AAMAS08-Workshop on Formal Models and Methods for Multi-Robot Systems LNCS. RoboCup 2005: Robot Soccer World Cup IX. Vol 4020/2006.*